Что ж, мы научили приложение создавать заметки, но было бы неплохо, чтобы их ещё можно было просматривать.

Просмотр заметки

Создайте новую Activity по аналогии с CreateNoteActivity: NoteActivity.

Разметка будет такой:

<?xml version="1.0" encoding="utf-8"?>

<android.support.design.widget.CoordinatorLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context="com.skillberg.notes.NoteActivity">

<android.support.design.widget.AppBarLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content">

<include layout="@layout/toolbar" />

</android.support.design.widget.AppBarLayout>

<ScrollView

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

app:layout\_behavior="@string/appbar\_scrolling\_view\_behavior">

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginEnd="16dp"

android:layout\_marginStart="16dp"

android:layout\_marginTop="16dp"

android:orientation="vertical">

<TextView

android:id="@+id/text\_tv"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:fontFamily="sans-serif"

android:textColor="@color/colorTextPrimary"

android:textSize="16sp" />

</LinearLayout>

</ScrollView>

</android.support.design.widget.CoordinatorLayout>

А Java-код таким:

public class NoteActivity extends AppCompatActivity {

private TextView noteTv;

@Override

protected void onCreate(@Nullable Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_note);

Toolbar toolbar = findViewById(R.id.toolbar);

setSupportActionBar(toolbar);

noteTv = findViewById(R.id.text\_tv);

}

}

Для того, чтобы Activity могла отобразить заметку, нужно передать в неё id заметки, после чего мы загрузим заметку из БД, используя этот id.

Мы можем передать id через Intent при вызове Activity, используя механизм **Extras**. Поскольку хорошим тоном является объявление ключей **Extra** в качестве констант, создадим такую константу:

Ещё нам понадобится поле, в котором мы будем хранить идентификатор заметки:

private long noteId;

Далее в конце onCreate() получим идентификатор заметки:

if (noteId == -1) {

finish();

}

Если идентификатор заметки равен -1, значит идентификатор не был передан. В таком случае мы просто завершаем Activity.

Теперь, точно так же, как и в MainActivity, будем использовать CursorLoader. Сделайте это сами.

Метод onCreateLoader() будет выглядеть несколько иначе:

@Override

public Loader<Cursor> onCreateLoader(int id, Bundle args) {

return new CursorLoader(

this, // Контекст

ContentUris.withAppendedId(NotesContract.Notes.URI, noteId), // URI

NotesContract.Notes.SINGLE\_PROJECTION, // Столбцы

null, // Параметры выборки

null, // Аргументы выборки

null // Сортировка по умолчанию

);

}

Изменились два параметра:

1. uri — теперь мы передаём туда **URI** конкретной заметки.
2. projection — мы изменили проекцию, поскольку при выборке конкретной заметки нам нужны все столбцы, а не ограниченный набор, как при выборке списка.

Всё остальное ContentResolver сделает за нас!

Обработка кликов по заметке

Мы с вами уже делали обработку кликов в адаптере. Давайте вы сделаете её сами, а я вам лишь подскажу, что именно нам нужно, и покажу интерфейс Listener-а:

public interface OnNoteClickListener {

void onNoteClick(long noteId);

}

Поскольку нам понадобится идентификатор заметки, нужно будет получить его из курсора. Сделайте это в методе onBindViewHolder по аналогии с другими столбцами, но не забудьте, что идентификатор — это не строка, а long.

После этого установите его в качестве тега во ViewHolder.

Не стесняйтесь подглядеть в тот код, что мы писали ранее, если не помните, как это делается.

Теперь нужно реализовать **Listener** в MainActivity и вызвать в нём NoteActivity:

@Override

public void onNoteClick(long noteId) {

Intent intent = new Intent(MainActivity.this, NoteActivity.class);

intent.putExtra(NoteActivity.EXTRA\_NOTE\_ID, noteId);

startActivity(intent);

}

};

В принципе, это ничем не отличается от обычного вызова Activity с использованием Intent, кроме того, что мы передаём идентификатор заметки в **Extras**.

На данный момент метод onLoadFinished() в NoteActivity должен выглядеть так:

@Override

public void onLoadFinished(Loader<Cursor> loader, Cursor cursor) {

Log.i("Test", "Load finished: " + cursor.getCount());

cursor.setNotificationUri(getContentResolver(), NotesContract.Notes.URI);

}

Если вы сейчас запустите приложение и кликните на заметку, то увидите в логах примерно следующее:

11-02 03:44:42.656 7284-7284/com.skillberg.notes I/Test: Load finished: 1

Это значит, что заметка загрузилась (учтите, что цифра должна равняться именно единице).

Теперь отобразим информацию о заметке — для этого получим данные из курсора точно так же, как и ранее в адаптере.

Создадим метод, который будет это делать:

private void displayNote(Cursor cursor) {

if (!cursor.moveToFirst()) {

// Если не получилось перейти к первой строке — завершаем Activity

finish();

}

String title = cursor.getString(cursor.getColumnIndexOrThrow(NotesContract.Notes.COLUMN\_TITLE));

String noteText = cursor.getString(cursor.getColumnIndexOrThrow(NotesContract.Notes.COLUMN\_NOTE));

setTitle(title);

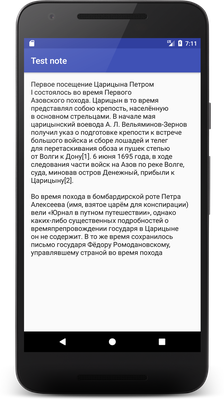
noteTv.setText(noteText);

}

И вызовем метод в конце onLoadFinished():

displayNote(cursor);

Откройте заметку, и вы увидите, что она отображается:

[](https://skillberg.com/media/uploads/2017/11/17/a3fe375a13df4730982d0450250f1208.png)Отображение заметки

Кнопка "назад" в Toolbar

Хорошим тоном является отображение кнопки **назад** (в виде стрелки) в Toolbar, чтобы пользователь мог вернуться на предыдущий экран не только физической кнопкой "назад".

Делается это достаточно просто.

Во-первых, в методе onCreate() сразу после установки тулбара добавьте такую строчку:

getSupportActionBar().setDisplayHomeAsUpEnabled(true);

Этой строчкой мы включили отображение кнопки. Осталось лишь обработать нажатие на эту кнопку и завершить Activity.

Эта кнопка, по сути, является элементом меню, поэтому обработать клик на неё мы можем в методе onOptionsItemSelected():

@Override

public boolean onOptionsItemSelected(MenuItem item) {

switch (item.getItemId()) {

case android.R.id.home:

finish();

return true;

default:

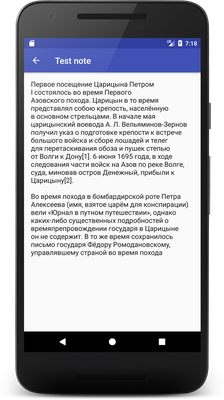
return super.onOptionsItemSelected(item);

}

}

android.R.id.home — идентификатор кнопки **назад**. По клику на неё мы просто завершаем Activity.

Запустите приложение и попробуйте:

[](https://skillberg.com/media/uploads/2017/11/17/2c30bc721a1e4a4bac3685c9c51361ad.png)Кнопка "назад" в Toolbar

Редактирование заметки

Для редактирования заметки мы будем использовать тот же экран, что и для создания.

Алгоритм таков: если в CreateNoteActivity был передан идентификатор заметки, то загружаем заметку по **ID**, отображаем её в полях редактирования. При сохранении обновляем заметку.

Если же идентификатор передан не был, то действуем как и ранее — создаём новую заметку.

Конечно же, пользователю нужно на что-то нажать, чтобы перейти к редактированию. Пусть это будет кнопка редактирования в Toolbar (такая же, как и кнопка сохранения на экране создания заметки).

Создайте кнопку самостоятельно по примеру из предыдущего урока.

Создайте метод editNote(), который будет вызван из обработчика клика на элемент меню:

private void editNote() {

Intent intent = new Intent(this, CreateNoteActivity.class);

intent.putExtra(CreateNoteActivity.EXTRA\_NOTE\_ID, noteId);

startActivity(intent);

}

Реализуйте получение заметки в CreateNoteActivity точно так же, как и в NoteActivity (с одним отличием — если **Extra** с идентификатором заметки не была передана, то мы не завершаем Activity):

public class CreateNoteActivity extends AppCompatActivity implements LoaderManager.LoaderCallbacks<Cursor> {

public static final String EXTRA\_NOTE\_ID = "note\_id";

// ...

private long noteId;

@Override

protected void onCreate(@Nullable Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_create\_note);

// ...

noteId = getIntent().getLongExtra(EXTRA\_NOTE\_ID, -1);

if (noteId != -1) {

getLoaderManager().initLoader(

0, // Идентификатор загрузчика

null, // Аргументы

this // Callback для событий загрузчика

);

}

}

// ...

@Override

public Loader<Cursor> onCreateLoader(int id, Bundle args) {

return new CursorLoader(

this, // Контекст

ContentUris.withAppendedId(NotesContract.Notes.URI, noteId), // URI

NotesContract.Notes.SINGLE\_PROJECTION, // Столбцы

null, // Параметры выборки

null, // Аргументы выборки

null // Сортировка по умолчанию

);

}

@Override

public void onLoadFinished(Loader<Cursor> loader, Cursor cursor) {

Log.i("Test", "Load finished: " + cursor.getCount());

cursor.setNotificationUri(getContentResolver(), NotesContract.Notes.URI);

displayNote(cursor);

}

@Override

public void onLoaderReset(Loader<Cursor> loader) {

}

private void displayNote(Cursor cursor) {

if (!cursor.moveToFirst()) {

// Если не получилось перейти к первой строке — завершаем Activity

finish();

}

String title = cursor.getString(cursor.getColumnIndexOrThrow(NotesContract.Notes.COLUMN\_TITLE));

String noteText = cursor.getString(cursor.getColumnIndexOrThrow(NotesContract.Notes.COLUMN\_NOTE));

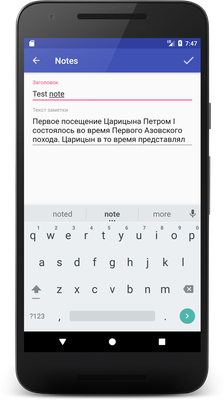
titleEt.setText(title);

textEt.setText(noteText);

}

}

Запустив приложение и нажав на кнопку редактирования, вы увидите, что всё успешно отображается:

[](https://skillberg.com/media/uploads/2017/11/17/9e02fc28c6ed48769178ee4c038492a6.png)Редактирование заметки

Кроме того, нужно изменить метод saveNote():

if (isCorrect) {

long currentTime = System.currentTimeMillis();

ContentValues contentValues = new ContentValues();

contentValues.put(NotesContract.Notes.COLUMN\_TITLE, title);

contentValues.put(NotesContract.Notes.COLUMN\_NOTE, text);

if (noteId == -1) {

contentValues.put(NotesContract.Notes.COLUMN\_CREATED\_TS, currentTime);

}

contentValues.put(NotesContract.Notes.COLUMN\_UPDATED\_TS, currentTime);

if (noteId == -1) {

getContentResolver().insert(NotesContract.Notes.URI, contentValues);

} else {

getContentResolver().update(ContentUris.withAppendedId(NotesContract.Notes.URI, noteId),

contentValues,

null,

null);

}

finish();

}

Если noteId не был задан, то сохраняем заметку как и раньше, если же он задан — то обновляем её. Само собой, если мы редактируем заметку, то оставляем дату создания той же, какой она и была.

Вы так же, наверное, помните, что в ContentProvider не был реализован метод update(). Давайте исправим это.

@Override

public int update(@NonNull Uri uri, @Nullable ContentValues contentValues,

@Nullable String selection, @Nullable String[] selectionArgs) {

SQLiteDatabase db = notesDbHelper.getWritableDatabase();

switch (URI\_MATCHER.match(uri)) {

case NOTE:

String id = uri.getLastPathSegment();

if (TextUtils.isEmpty(selection)) {

selection = NotesContract.Notes.\_ID + " = ?";

selectionArgs = new String[]{id};

} else {

selection = selection + " AND " + NotesContract.Notes.\_ID + " = ?";

String[] newSelectionArgs = new String[selectionArgs.length + 1];

System.arraycopy(selectionArgs, 0, newSelectionArgs, 0, selectionArgs.length);

newSelectionArgs[newSelectionArgs.length - 1] = id;

selectionArgs = newSelectionArgs;

}

int rowsUpdated = db.update(NotesContract.Notes.TABLE\_NAME, contentValues, selection, selectionArgs);

getContext().getContentResolver().notifyChange(uri, null);

return rowsUpdated;

}

return 0;

}

По сути, получилась комбинация методов query() и insert(). Единственное существенное отличие — то, что мы возвращаем не URI изменённого контента, а количество изменённых строк.

Теперь попробуйте отредактировать заметку, и увидите, что она успешно отредактировалась :)

ПРИМЕРЫ КОДА

[ИСХОДНЫЙ КОД УРОКА](https://github.com/Skillberg/NotesAndroid/tree/lesson-25)

ДОМАШНЕЕ ЗАДАНИЕ

Добавьте кнопку "Назад" на экран создания заметки.